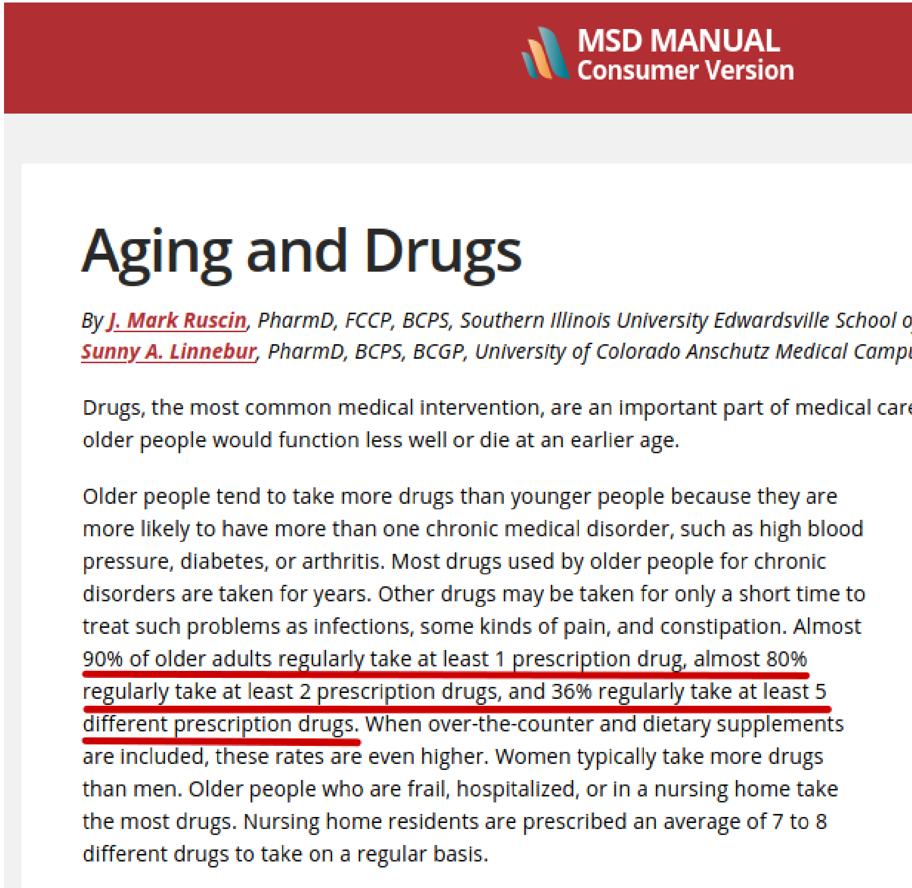
Tablet/Pill/Capsule recognition using Deep Learning

Name: Ashish.V.Nair

Purpose

- We live in a world were post a certain age, unlike older times, it is confirmed that we will mostly be sufferring from a few diseases and we have to spend the rest of our lives depending on a few medical pills to maintain our health.
- When monitorred, we would notice that an old person would be consuming around 1-5 of such medical pills per day. Over the years, with their memory becoming weaker, or due to changes in subscribed medical pills, it is often noticed that these old people accidentally/unknowingly consume the wrong medical pills.
- This is extremely hazardous to them and sometimes causes irreplaceable damages or at bare minimum, this leads to further deteriotion of their health.
- This can be avoided by allowing an intelligent Pill dispencer help the old people procure their timely, subscribed medical pills.

INFO SOURCE : https://www.msdmanuals.com



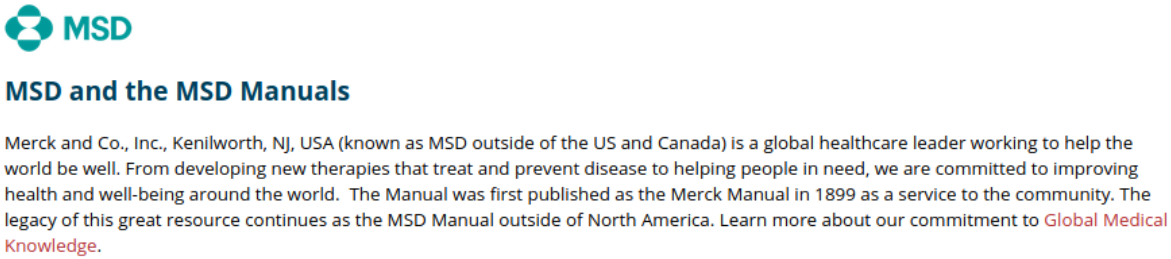
MSD MANUAL
Consumer Version

Aging and Drugs

By J. Mark Ruscin, PharmD, FCCP, BCPS, Southern Illinois University Edwardsville School of Pharmacy;
Sunny A. Linnebur, PharmD, BCPS, BCGP, University of Colorado Anschutz Medical Campus

Drugs, the most common medical intervention, are an important part of medical care for older people. Without drugs, many older people would function less well or die at an earlier age.

Older people tend to take more drugs than younger people because they are more likely to have more than one chronic medical disorder, such as high blood pressure, diabetes, or arthritis. Most drugs used by older people for chronic disorders are taken for years. Other drugs may be taken for only a short time to treat such problems as infections, some kinds of pain, and constipation. Almost 90% of older adults regularly take at least 1 prescription drug, almost 80% regularly take at least 2 prescription drugs, and 36% regularly take at least 5 different prescription drugs. When over-the-counter and dietary supplements are included, these rates are even higher. Women typically take more drugs than men. Older people who are frail, hospitalized, or in a nursing home take the most drugs. Nursing home residents are prescribed an average of 7 to 8 different drugs to take on a regular basis.



MSD

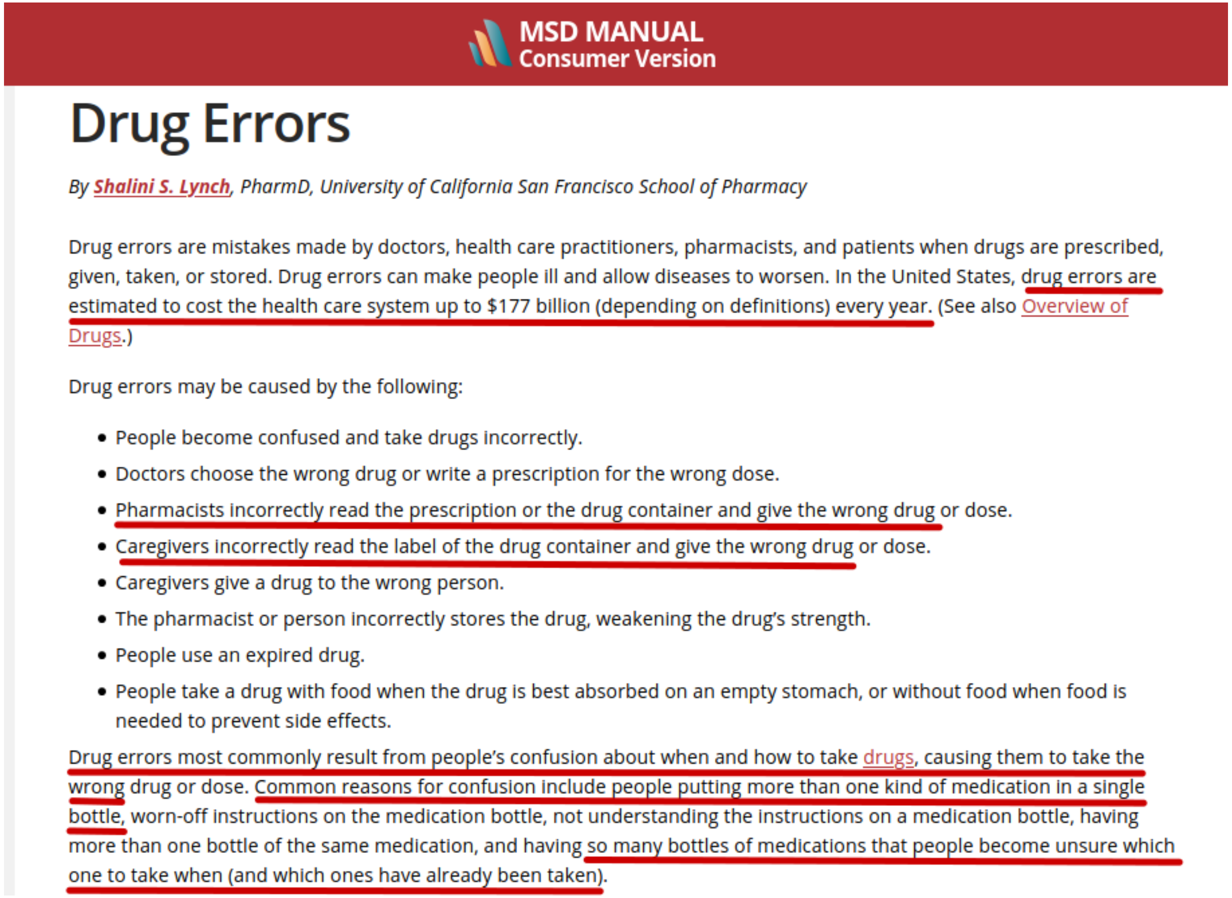
MSD and the MSD Manuals

Merck and Co., Inc., Kenilworth, NJ, USA (known as MSD outside of the US and Canada) is a global healthcare leader working to help the world be well. From developing new therapies that treat and prevent disease to helping people in need, we are committed to improving health and well-being around the world. The Manual was first published as the Merck Manual in 1899 as a service to the community. The legacy of this great resource continues as the MSD Manual outside of North America. Learn more about our commitment to Global Medical Knowledge.

Note:

As per the information provided / highlighted we could confirm that elderly people consume around 1-5 medical pills regularly.

INFO SOURCE : https://www.msdmanuals.com



MSD MANUAL
Consumer Version

Drug Errors

By *Shalini S. Lynch*, *PharmD, University of California San Francisco School of Pharmacy*

Drug errors are mistakes made by doctors, health care practitioners, pharmacists, and patients when drugs are prescribed, given, taken, or stored. Drug errors can make people ill and allow diseases to worsen. In the United States, drug errors are estimated to cost the health care system up to $177 billion (depending on definitions) every year. (See also Overview of Drugs.)

Drug errors may be caused by the following:

- People become confused and take drugs incorrectly.
- Doctors choose the wrong drug or write a prescription for the wrong dose.
- Pharmacists incorrectly read the prescription or the drug container and give the wrong drug or dose.
- Caregivers incorrectly read the label of the drug container and give the wrong drug or dose.
- Caregivers give a drug to the wrong person.
- The pharmacist or person incorrectly stores the drug, weakening the drug's strength.
- People use an expired drug.
- People take a drug with food when the drug is best absorbed on an empty stomach, or without food when food is needed to prevent side effects.

Drug errors most commonly result from people's confusion about when and how to take drugs, causing them to take the wrong drug or dose. Common reasons for confusion include people putting more than one kind of medication in a single bottle, worn-off instructions on the medication bottle, not understanding the instructions on a medication bottle, having more than one bottle of the same medication, and having so many bottles of medications that people become unsure which one to take when (and which ones have already been taken).

Note:

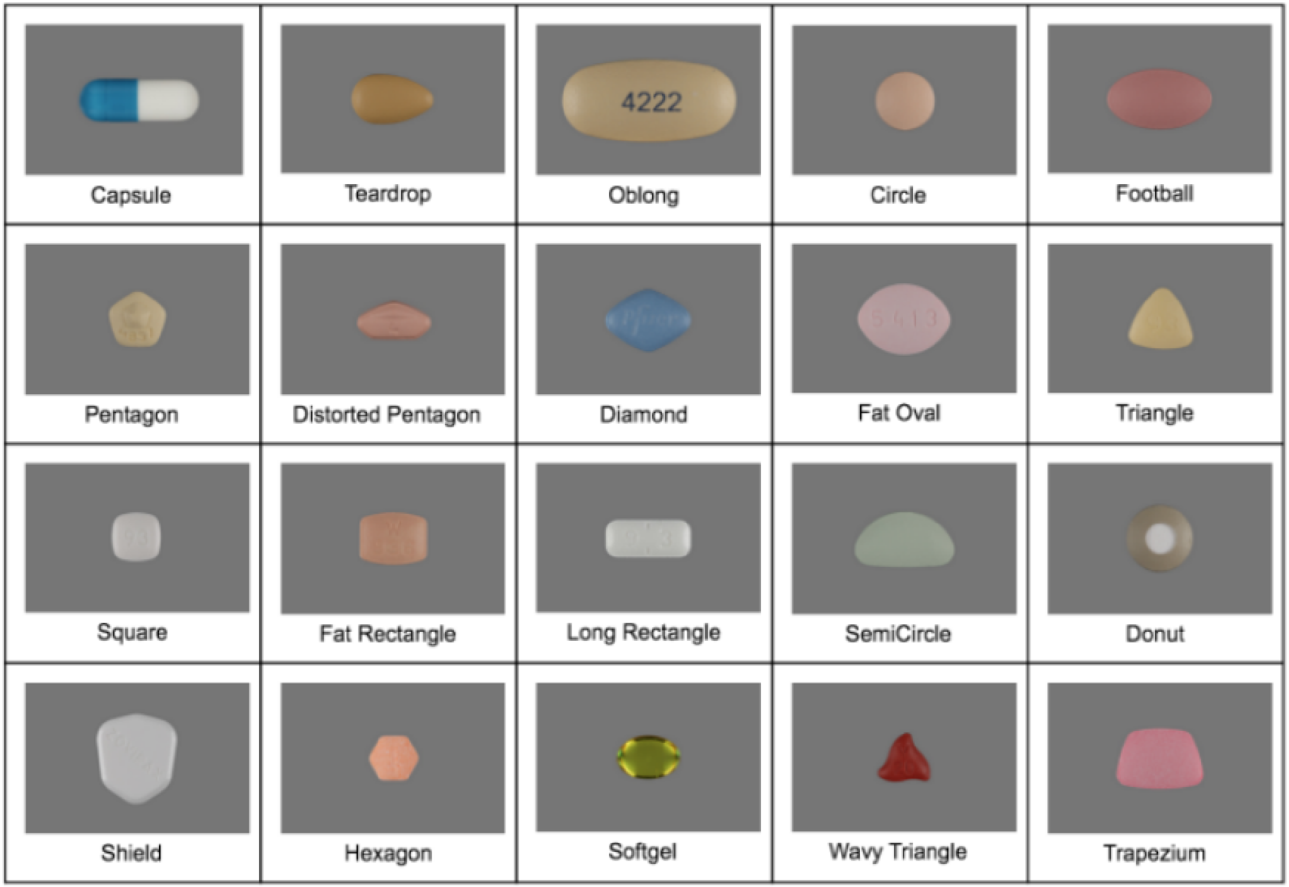
As per the information provided / highlighted we could confirm that pill dispensing errors are common and fatal.

Methods to Resolve Pill Dispencing Issue

Pill recognition by :

- Classification of Shape
  - This is the first criteria towards identifying any pill. Example Digene is circular, Crocin is oblong in shape.
- Recognition of Color [not covered in this Project]
  - Pill Dispenser could have gray background, this will help us to check the color of Pill we need to check.
- OCR of text on Pill [not covered in this Project]
  - Pills have text/code printed on them, this could be crossverified against database to confirm identity of the selected pill.

Pill recognition by classification of **SHAPE**

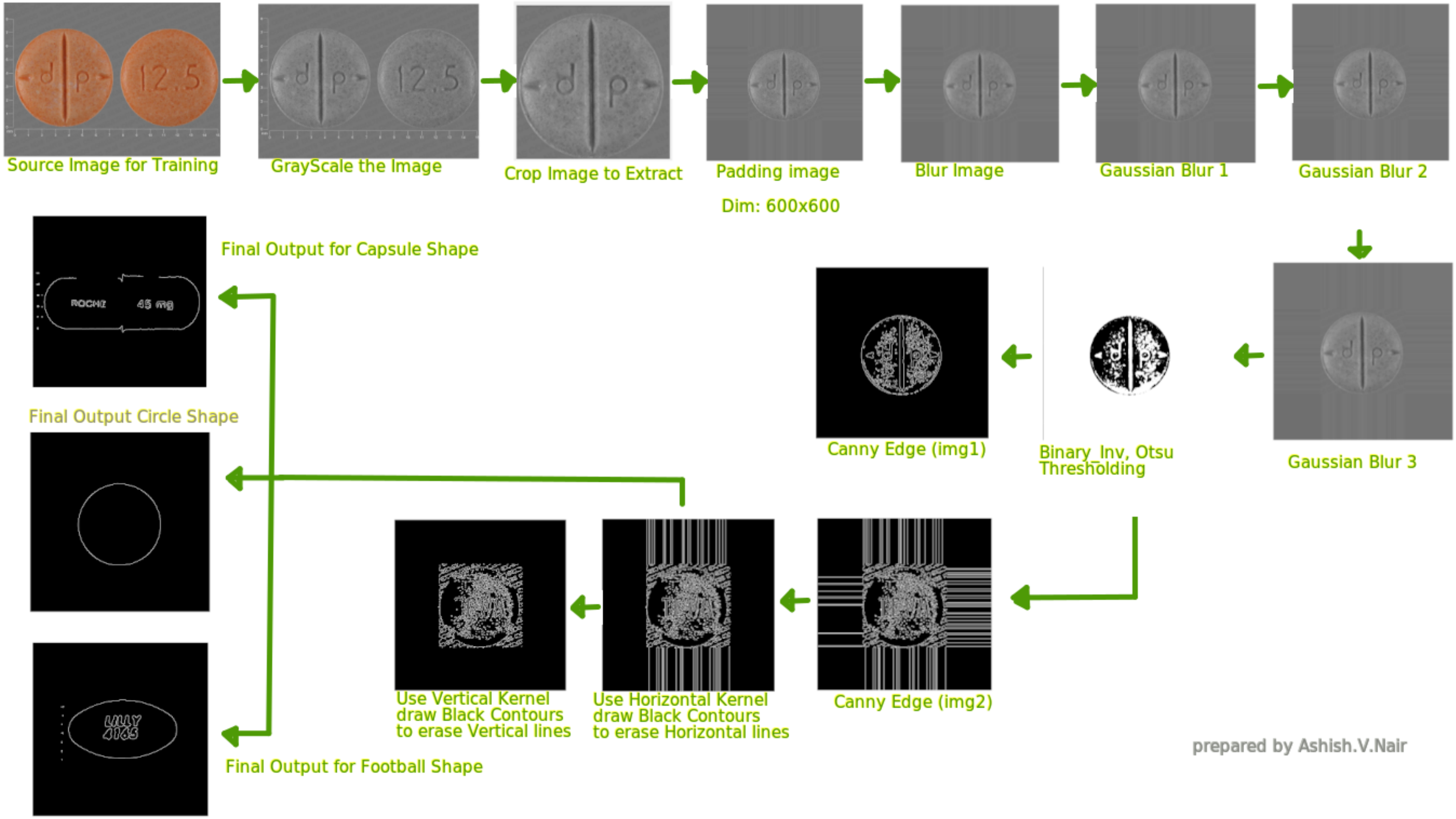


Note:
Different shapes of medical pills available in the market.

Pill recognition by classification of SHAPE

- Data Source: https://www.nlm.nih.gov/databases/download/pill_image.html
- From Data Source we would be using only following pill shapes for training and classification;
  - Capsule
  - Circle
  - Football

PreProcessing for Train and Test Dataset >



Source Image for Training

GrayScale the Image

Crop Image to Extract

Padding image

Dim: 600x600

Blur Image

Gaussian Blur 1

Gaussian Blur 2

Gaussian Blur 3

Binary_Inv, Otsu Thresholding

Canny Edge (img1)

Canny Edge (img2)

Use Horizontal Kernel draw Black Contours to erase Horizontal lines

Use Vertical Kernel draw Black Contours to erase Vertical lines

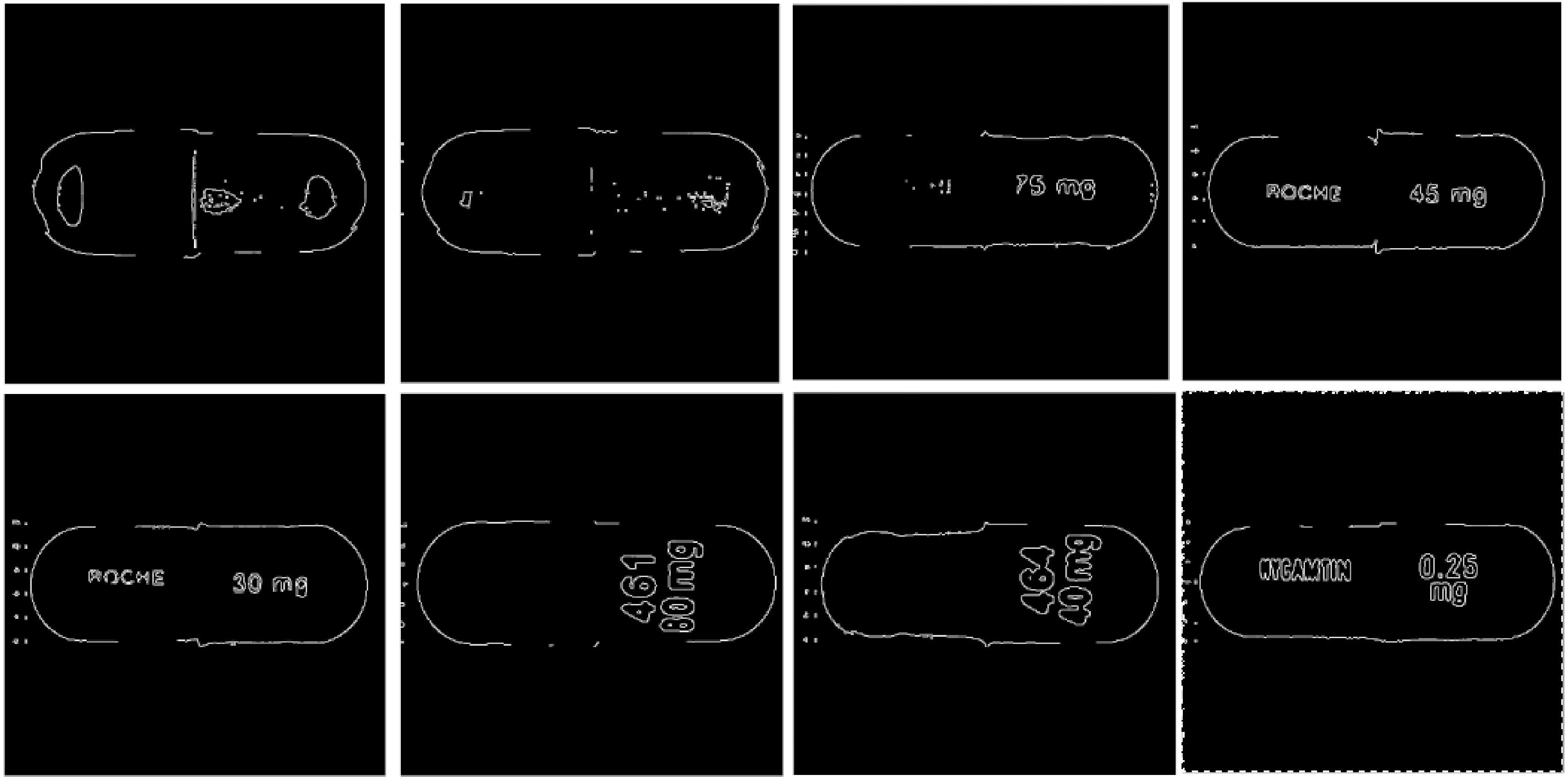
Final Output for Capsule Shape

Final Output Circle Shape

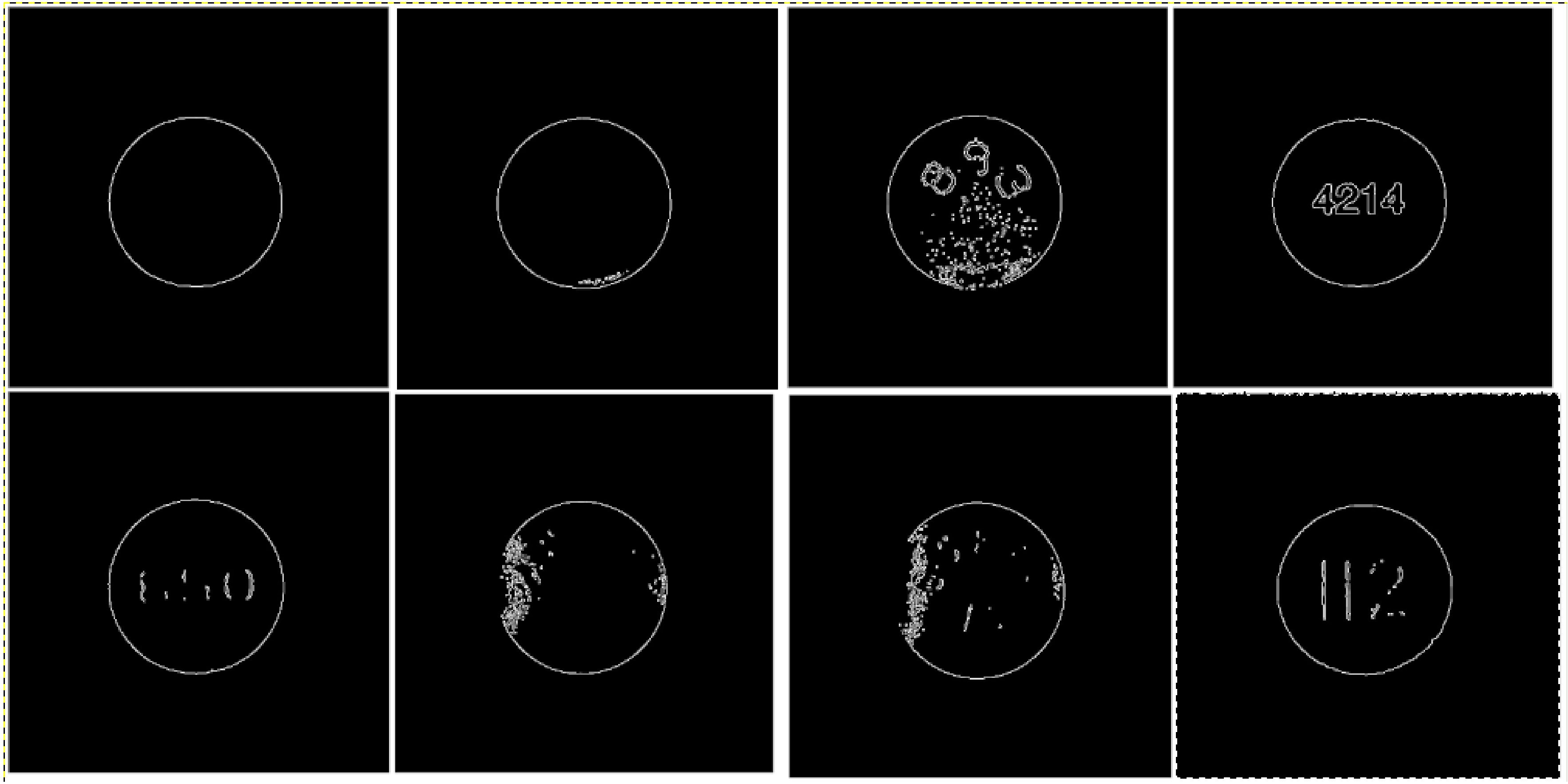
Final Output for Football Shape

prepared by Ashish.V.Nair

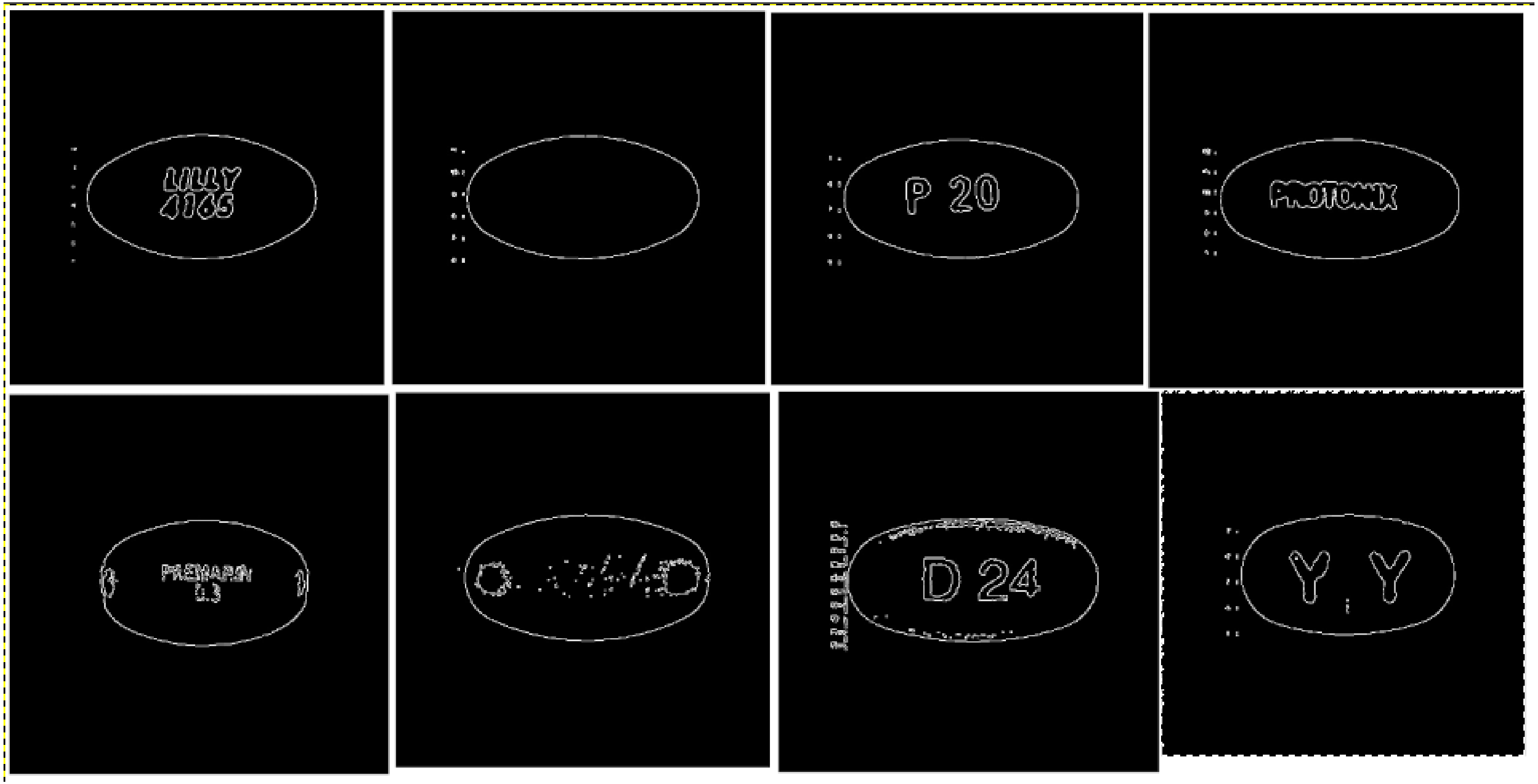
PreProcessed capsule shaped Pill (600x600 [px])



PreProcessed circle shaped Pill (600x600 [px])



PreProcessed football shaped Pill (600x600 [px])



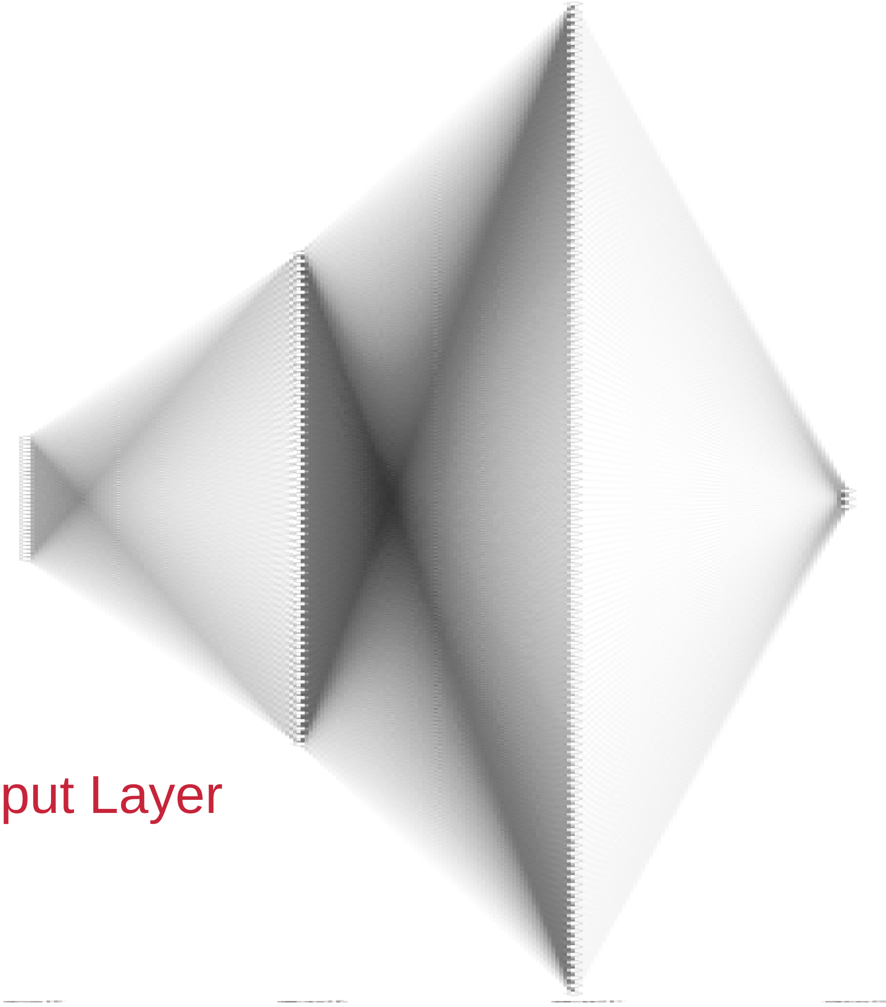
Keras Convolution 2D Layers

Layer 1: Input Layer
32 Neurons

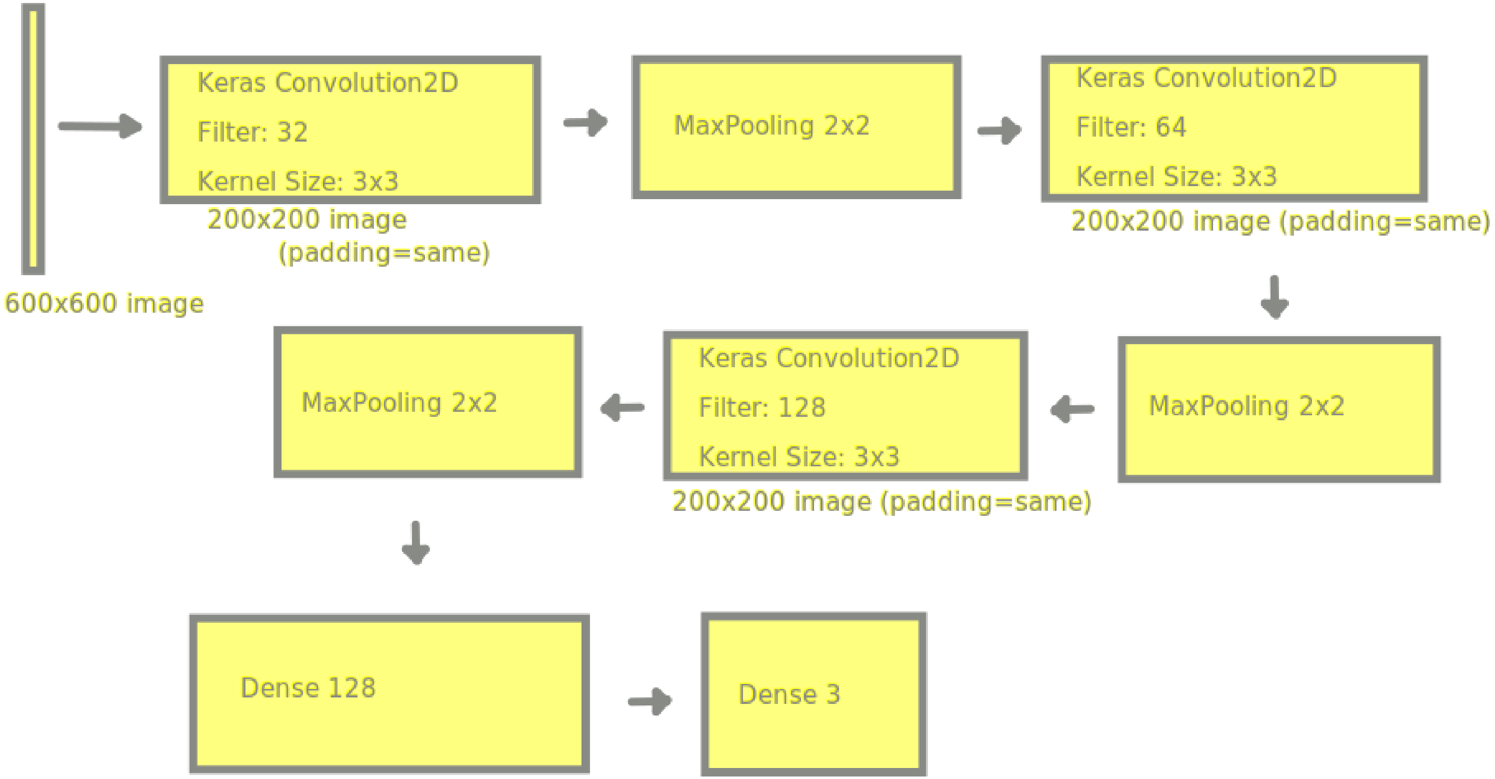
Layer 2: Hidden Layer
64 Neurons

Layer 3: Hidden Layer
128 Neurons

Layer 4: Output Layer
3 Neurons



Keras Convolution2D (architectire utilized)



600x600 image

Keras Convolution2D

Filter: 32

Kernel Size: 3x3

200x200 image (padding=same)

MaxPooling 2x2

Keras Convolution2D

Filter: 64

Kernel Size: 3x3

200x200 image (padding=same)

MaxPooling 2x2

Keras Convolution2D

Filter: 128

Kernel Size: 3x3

200x200 image (padding=same)

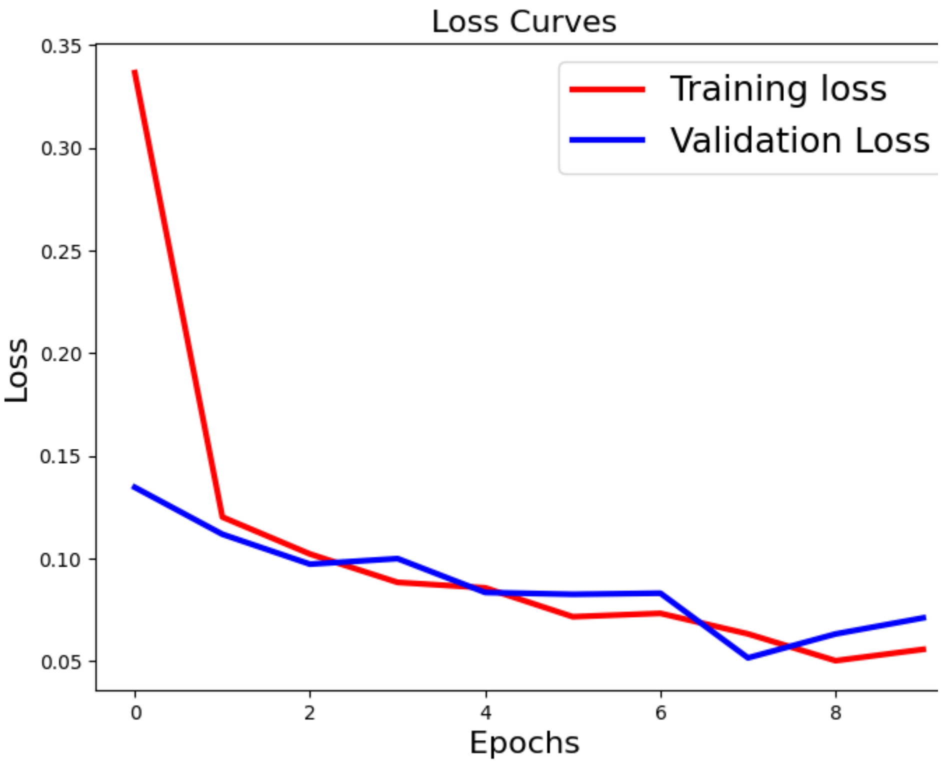
MaxPooling 2x2

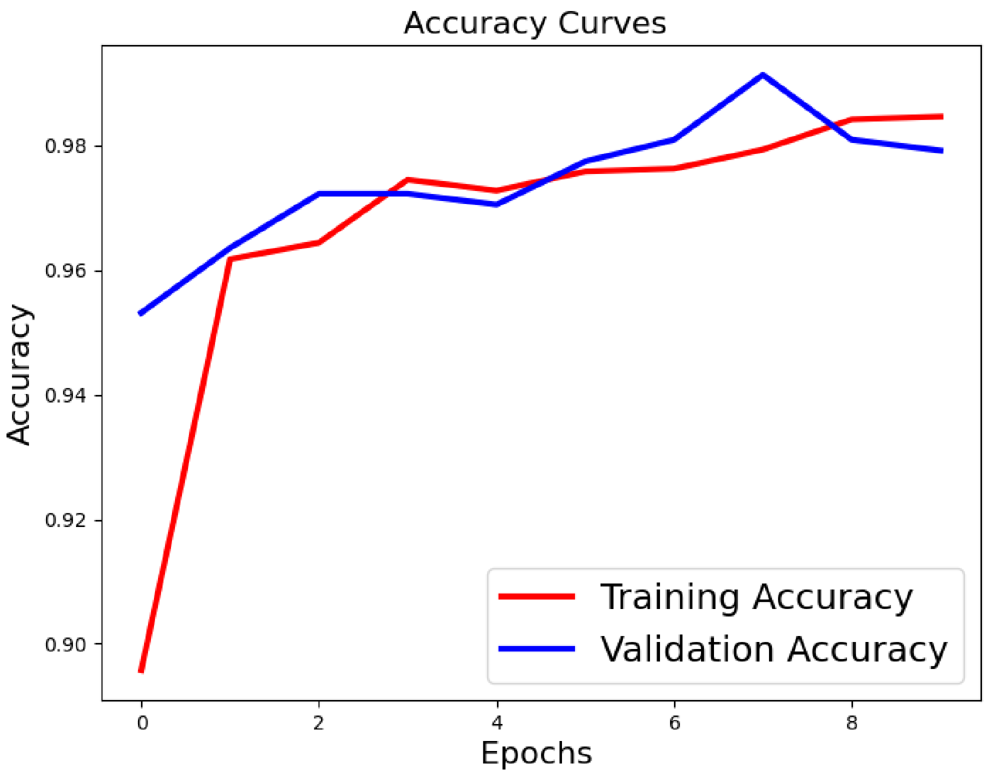
Dense 128

Dense 3

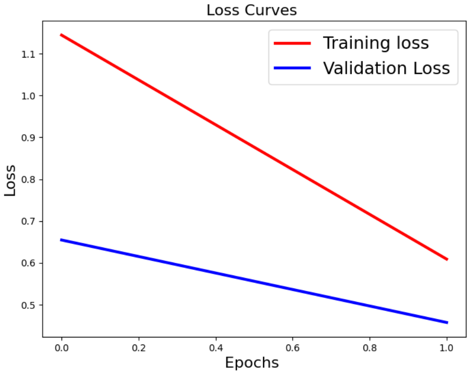
prepared by Ashish.V.Nair

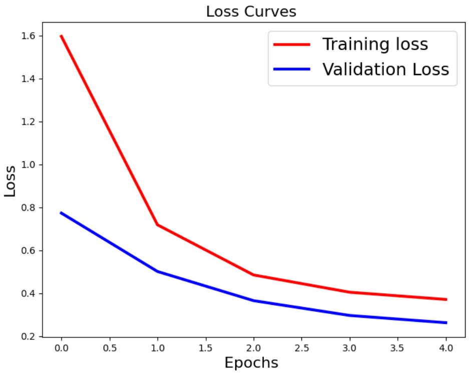
(Validation / Training) [Loss,Accuracy] vs [Number of Epochs]

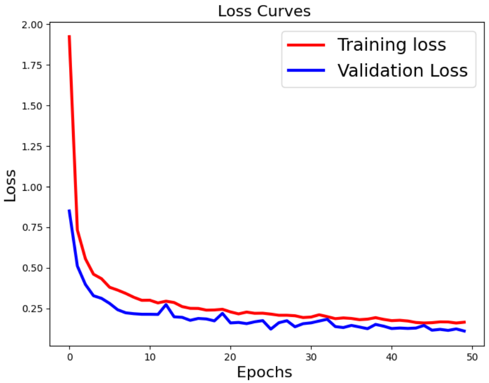


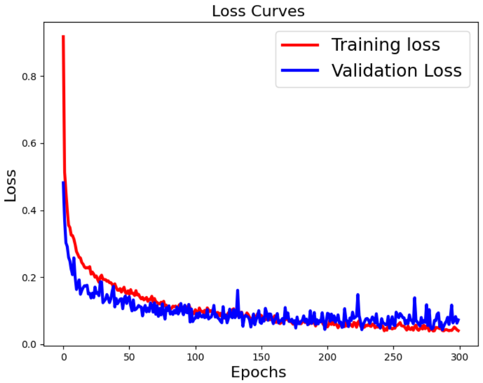


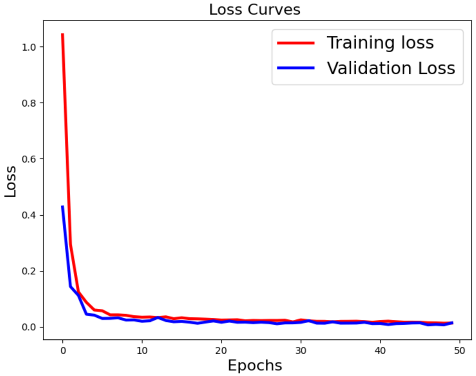
Loss reduction during Model development (previous graphs)

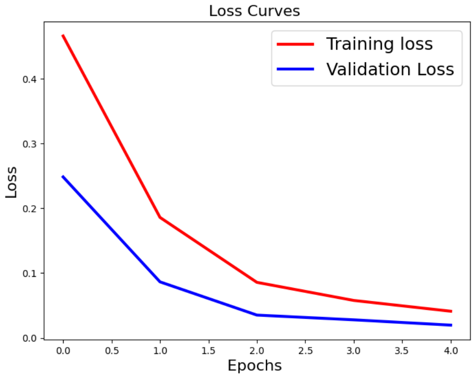








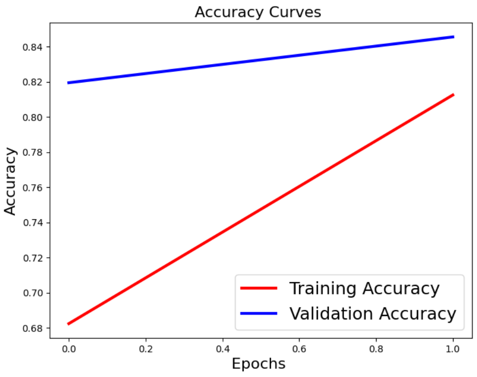


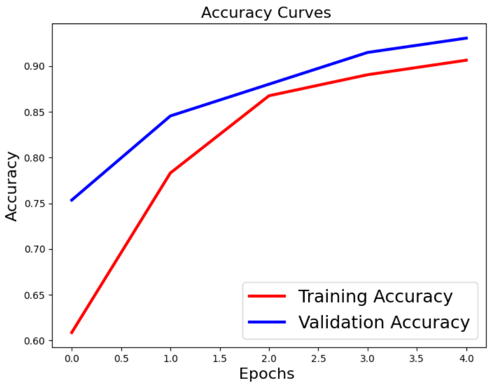


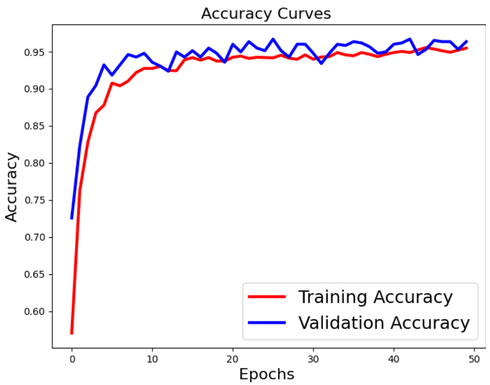
Experimentally earned knowledge:

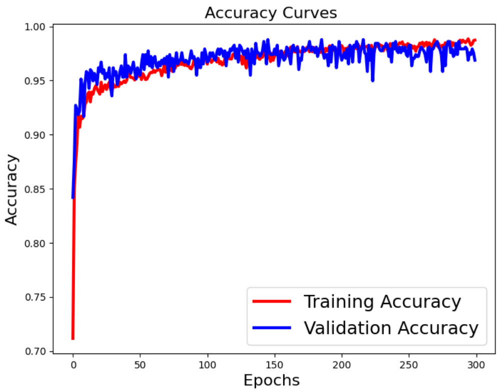
1. PreProcessing decreases loss.
2. Higher Epochs decreases loss.
3. Better preprocessing can reduce Epochs required.
4. Better model if difference between training and validation losses are less.
5. Time consumed per Epoch reduces when better preprocessed images used to train model.

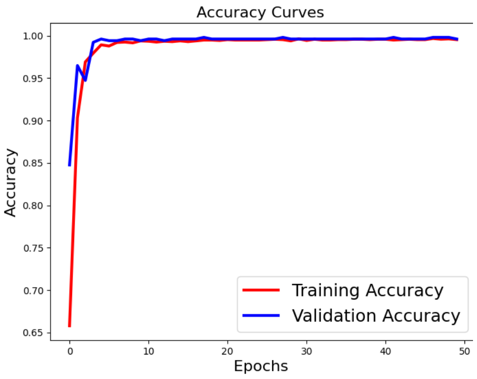
Accuracy incrementation during Model development (previous graphs)

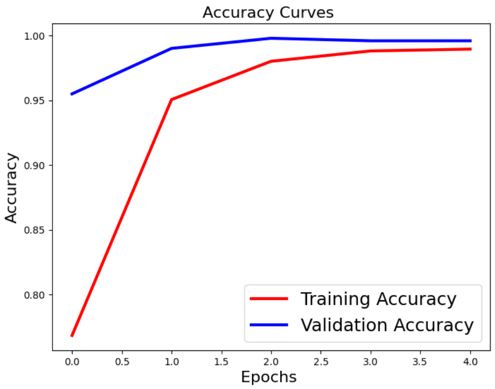












Experimentally earned knowledge:

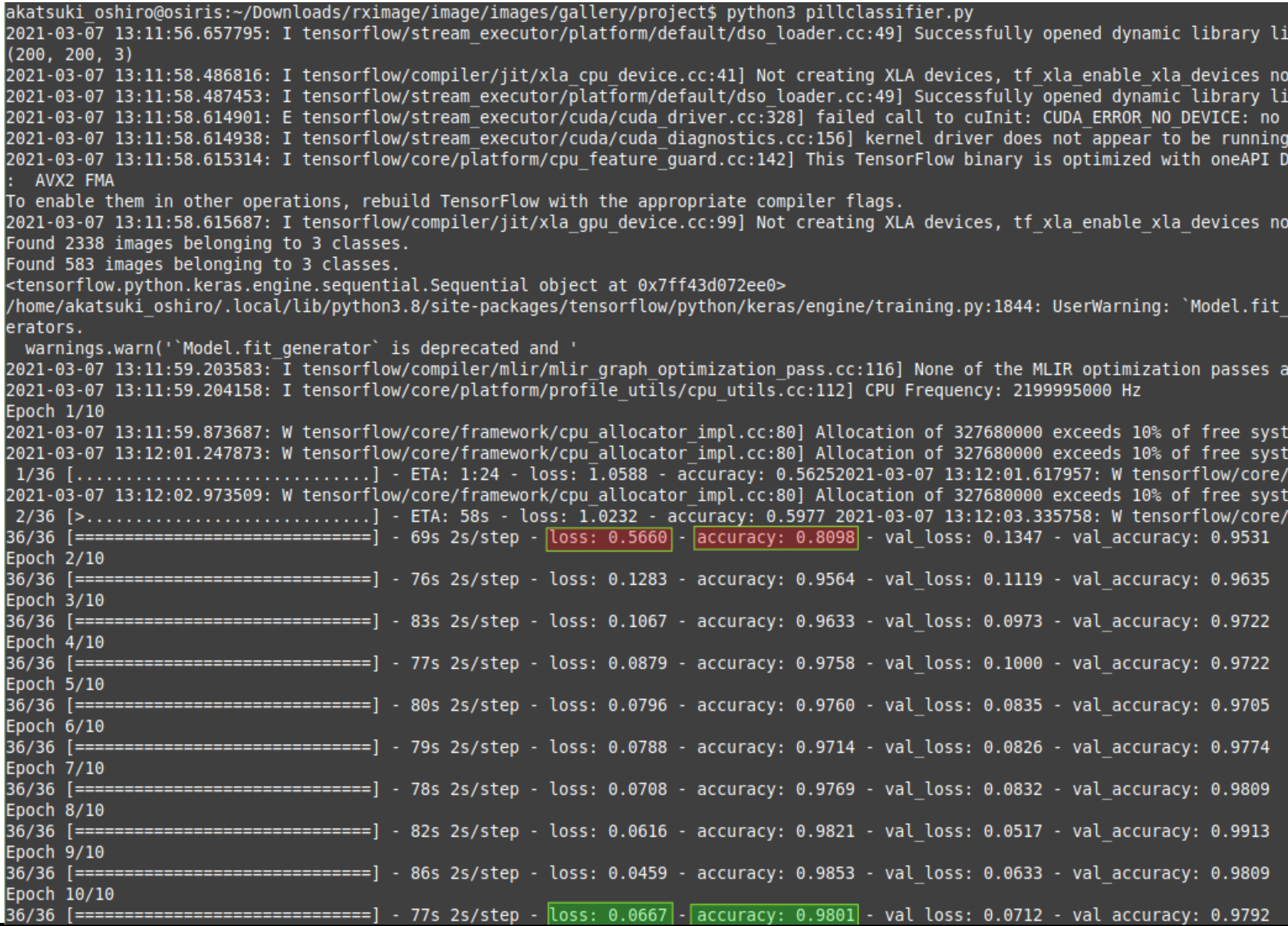
1. PreProcessing increases accuracy.
2. Higher Epochs increases accuracy.
3. Better preprocessing can reduce Epochs required.
4. Better model if difference between training and validation accuracy is less.
5. Time consumed per Epoch reduces when better preprocessed images used to train model.

Model Details

Model Name: fmodel.h5
Model Size: 80Mb(approx.)

3 Class classification model

Note:
Image to the right shows model development where we can see the training loss decreasing and accuracy increasing.

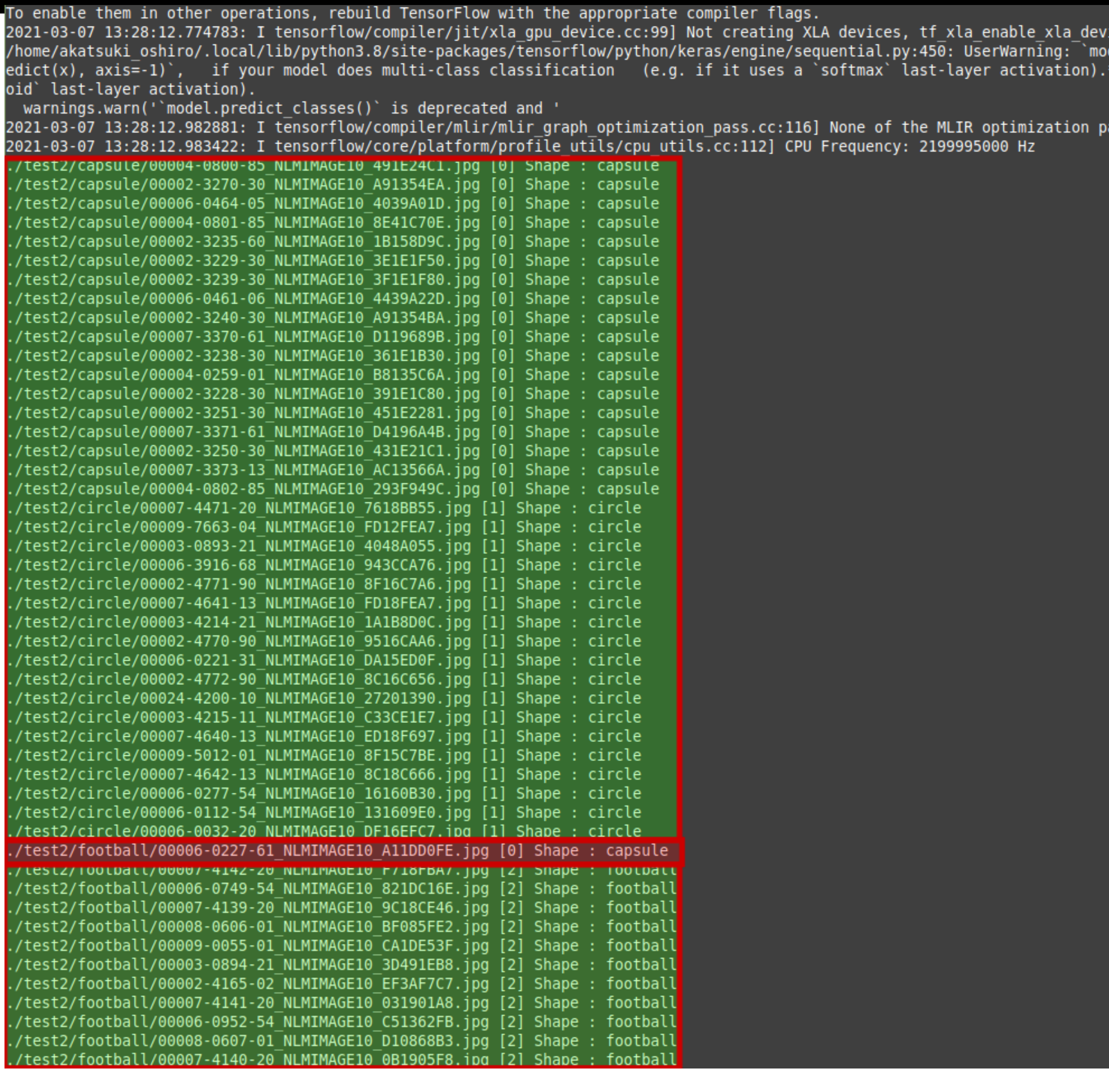


```
akatsuki_oshiro@osiris:~/Downloads/rximage/image/images/gallery/project$ python3 pillclassifier.py
2021-03-07 13:11:56.657795: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library li
(200, 200, 3)
2021-03-07 13:11:58.486816: I tensorflow/compiler/jit/xla_cpu_device.cc:41] Not creating XLA devices, tf_xla_enable_xla_devices no
2021-03-07 13:11:58.487453: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library li
2021-03-07 13:11:58.614901: E tensorflow/stream_executor/cuda/cuda_driver.cc:328] failed call to cuInit: CUDA_ERROR_NO_DEVICE: no
2021-03-07 13:11:58.614938: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running
2021-03-07 13:11:58.615314: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI D
:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-03-07 13:11:58.615687: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_devices no
Found 2338 images belonging to 3 classes.
Found 583 images belonging to 3 classes.
<tensorflow.python.keras.engine.sequential.Sequential object at 0x7ff43d072ee0>
/home/akatsuki_oshiro/.local/lib/python3.8/site-packages/tensorflow/python/keras/engine/training.py:1844: UserWarning: `Model.fit_
erators.
  warnings.warn('`Model.fit_generator` is deprecated and '
2021-03-07 13:11:59.203583: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes a
2021-03-07 13:11:59.204158: I tensorflow/core/platform/profile_utils/cpu_utils.cc:112] CPU Frequency: 2199995000 Hz
Epoch 1/10
2021-03-07 13:11:59.873687: W tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of 327680000 exceeds 10% of free syst
2021-03-07 13:12:01.247873: W tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of 327680000 exceeds 10% of free syst
 1/36 [..............................] - ETA: 1:24 - loss: 1.0588 - accuracy: 0.56252021-03-07 13:12:01.617957: W tensorflow/core/
2021-03-07 13:12:02.973509: W tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of 327680000 exceeds 10% of free syst
 2/36 [>.............................] - ETA: 58s - loss: 1.0232 - accuracy: 0.5977 2021-03-07 13:12:03.335758: W tensorflow/core/
36/36 [==============================] - 69s 2s/step - loss: 0.5660 - accuracy: 0.8098 - val_loss: 0.1347 - val_accuracy: 0.9531
Epoch 2/10
36/36 [==============================] - 76s 2s/step - loss: 0.1283 - accuracy: 0.9564 - val_loss: 0.1119 - val_accuracy: 0.9635
Epoch 3/10
36/36 [==============================] - 83s 2s/step - loss: 0.1067 - accuracy: 0.9633 - val_loss: 0.0973 - val_accuracy: 0.9722
Epoch 4/10
36/36 [==============================] - 77s 2s/step - loss: 0.0879 - accuracy: 0.9758 - val_loss: 0.1000 - val_accuracy: 0.9722
Epoch 5/10
36/36 [==============================] - 80s 2s/step - loss: 0.0796 - accuracy: 0.9760 - val_loss: 0.0835 - val_accuracy: 0.9705
Epoch 6/10
36/36 [==============================] - 79s 2s/step - loss: 0.0788 - accuracy: 0.9714 - val_loss: 0.0826 - val_accuracy: 0.9774
Epoch 7/10
36/36 [==============================] - 78s 2s/step - loss: 0.0708 - accuracy: 0.9769 - val_loss: 0.0832 - val_accuracy: 0.9809
Epoch 8/10
36/36 [==============================] - 82s 2s/step - loss: 0.0616 - accuracy: 0.9821 - val_loss: 0.0517 - val_accuracy: 0.9913
Epoch 9/10
36/36 [==============================] - 86s 2s/step - loss: 0.0459 - accuracy: 0.9853 - val_loss: 0.0633 - val_accuracy: 0.9809
Epoch 10/10
36/36 [==============================] - 77s 2s/step - loss: 0.0667 - accuracy: 0.9801 - val_loss: 0.0712 - val_accuracy: 0.9792
```

Working Output

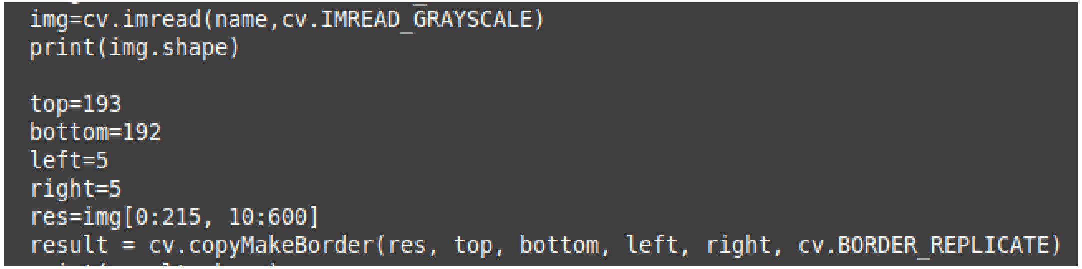
Out of around **50+** test images, only **1** was false output.

Ran it in loop to verify filename and identified shape per image.



```
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-03-07 13:28:12.774783: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_dev
/home/akatsuki_oshiro/.local/lib/python3.8/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarning: `mo
edict(x), axis=-1)`,   if your model does multi-class classification   (e.g. if it uses a `softmax` last-layer activation).
oid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
2021-03-07 13:28:12.982881: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization p
2021-03-07 13:28:12.983422: I tensorflow/core/platform/profile_utils/cpu_utils.cc:112] CPU Frequency: 2199995000 Hz
./test2/capsule/00004-0800-85_NLMIMAGE10_491E24C1.jpg [0] Shape : capsule
./test2/capsule/00002-3270-30_NLMIMAGE10_A91354EA.jpg [0] Shape : capsule
./test2/capsule/00006-0464-05_NLMIMAGE10_4039A01D.jpg [0] Shape : capsule
./test2/capsule/00004-0801-85_NLMIMAGE10_8E41C70E.jpg [0] Shape : capsule
./test2/capsule/00002-3235-60_NLMIMAGE10_1B158D9C.jpg [0] Shape : capsule
./test2/capsule/00002-3229-30_NLMIMAGE10_3E1E1F50.jpg [0] Shape : capsule
./test2/capsule/00002-3239-30_NLMIMAGE10_3F1E1F80.jpg [0] Shape : capsule
./test2/capsule/00006-0461-06_NLMIMAGE10_4439A22D.jpg [0] Shape : capsule
./test2/capsule/00002-3240-30_NLMIMAGE10_A91354BA.jpg [0] Shape : capsule
./test2/capsule/00007-3370-61_NLMIMAGE10_D119689B.jpg [0] Shape : capsule
./test2/capsule/00002-3238-30_NLMIMAGE10_361E1B30.jpg [0] Shape : capsule
./test2/capsule/00004-0259-01_NLMIMAGE10_B8135C6A.jpg [0] Shape : capsule
./test2/capsule/00002-3228-30_NLMIMAGE10_391E1C80.jpg [0] Shape : capsule
./test2/capsule/00002-3251-30_NLMIMAGE10_451E2281.jpg [0] Shape : capsule
./test2/capsule/00007-3371-61_NLMIMAGE10_D4196A4B.jpg [0] Shape : capsule
./test2/capsule/00002-3250-30_NLMIMAGE10_431E21C1.jpg [0] Shape : capsule
./test2/capsule/00007-3373-13_NLMIMAGE10_AC13566A.jpg [0] Shape : capsule
./test2/capsule/00004-0802-85_NLMIMAGE10_293F949C.jpg [0] Shape : capsule
./test2/circle/00007-4471-20_NLMIMAGE10_7618BB55.jpg [1] Shape : circle
./test2/circle/00009-7663-04_NLMIMAGE10_FD12FEA7.jpg [1] Shape : circle
./test2/circle/00003-0893-21_NLMIMAGE10_4048A055.jpg [1] Shape : circle
./test2/circle/00006-3916-68_NLMIMAGE10_943CCA76.jpg [1] Shape : circle
./test2/circle/00002-4771-90_NLMIMAGE10_8F16C7A6.jpg [1] Shape : circle
./test2/circle/00007-4641-13_NLMIMAGE10_FD18FEA7.jpg [1] Shape : circle
./test2/circle/00003-4214-21_NLMIMAGE10_1A1B8D0C.jpg [1] Shape : circle
./test2/circle/00002-4770-90_NLMIMAGE10_9516CAA6.jpg [1] Shape : circle
./test2/circle/00006-0221-31_NLMIMAGE10_DA15ED0F.jpg [1] Shape : circle
./test2/circle/00002-4772-90_NLMIMAGE10_8C16C656.jpg [1] Shape : circle
./test2/circle/00024-4200-10_NLMIMAGE10_27201390.jpg [1] Shape : circle
./test2/circle/00003-4215-11_NLMIMAGE10_C33CE1E7.jpg [1] Shape : circle
./test2/circle/00007-4640-13_NLMIMAGE10_ED18F697.jpg [1] Shape : circle
./test2/circle/00009-5012-01_NLMIMAGE10_8F15C7BE.jpg [1] Shape : circle
./test2/circle/00007-4642-13_NLMIMAGE10_8C18C666.jpg [1] Shape : circle
./test2/circle/00006-0277-54_NLMIMAGE10_16160B30.jpg [1] Shape : circle
./test2/circle/00006-0112-54_NLMIMAGE10_131609E0.jpg [1] Shape : circle
./test2/circle/00006-0032-20_NLMIMAGE10_DF16EFC7.jpg [1] Shape : circle
./test2/football/00006-0227-61_NLMIMAGE10_A11DD0FE.jpg [0] Shape : capsule
./test2/football/00007-4142-20_NLMIMAGE10_F718FBA7.jpg [2] Shape : football
./test2/football/00006-0749-54_NLMIMAGE10_821DC16E.jpg [2] Shape : football
./test2/football/00007-4139-20_NLMIMAGE10_9C18CE46.jpg [2] Shape : football
./test2/football/00008-0606-01_NLMIMAGE10_BF085FE2.jpg [2] Shape : football
./test2/football/00009-0055-01_NLMIMAGE10_CA1DE53F.jpg [2] Shape : football
./test2/football/00003-0894-21_NLMIMAGE10_3D491EB8.jpg [2] Shape : football
./test2/football/00002-4165-02_NLMIMAGE10_EF3AF7C7.jpg [2] Shape : football
./test2/football/00007-4141-20_NLMIMAGE10_031901A8.jpg [2] Shape : football
./test2/football/00006-0952-54_NLMIMAGE10_C51362FB.jpg [2] Shape : football
./test2/football/00008-0607-01_NLMIMAGE10_D10868B3.jpg [2] Shape : football
./test2/football/00007-4140-20_NLMIMAGE10_0B1905F8.jpg [2] Shape : football
```

Important Code Parts



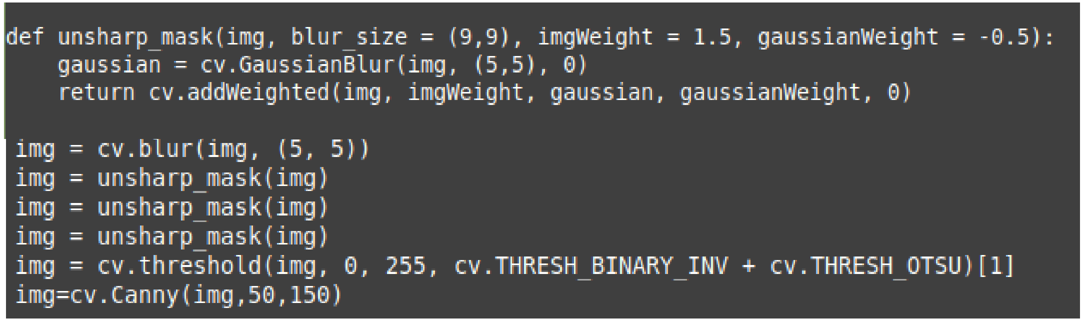
```
img=cv.imread(name,cv.IMREAD_GRAYSCALE)
print(img.shape)

top=193
bottom=192
left=5
right=5
res=img[0:215, 10:600]
result = cv.copyMakeBorder(res, top, bottom, left, right, cv.BORDER_REPLICATE)
```

Python Code Part:

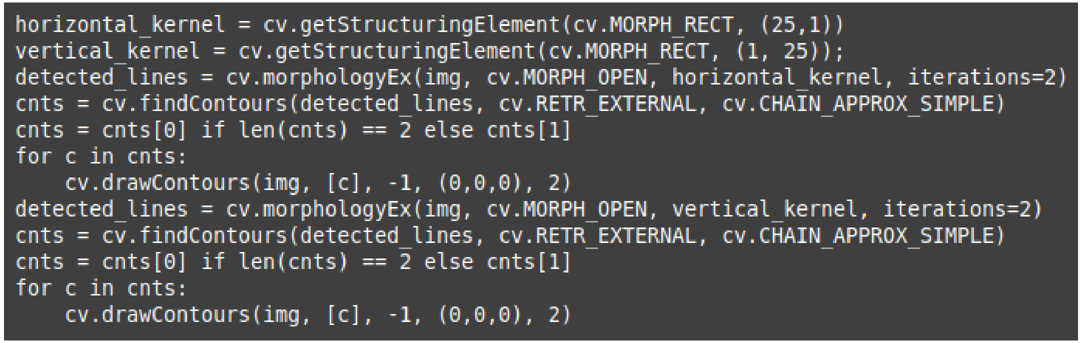
1. Read image as GrayScale.
2. Crop image (this is code for Capsule shape)
3. Make replicating border to make image 600x600.

Python Code Part:

1. Blur the image.
2. Gaussian Blur the image multiple times.
3. Thresholding via Binary_Inv and Otsu.
4. Perform Canny edge detection.



```
def unsharp_mask(img, blur_size = (9,9), imgWeight = 1.5, gaussianWeight = -0.5):
    gaussian = cv.GaussianBlur(img, (5,5), 0)
    return cv.addWeighted(img, imgWeight, gaussian, gaussianWeight, 0)

img = cv.blur(img, (5, 5))
img = unsharp_mask(img)
img = unsharp_mask(img)
img = unsharp_mask(img)
img = cv.threshold(img, 0, 255, cv.THRESH_BINARY_INV + cv.THRESH_OTSU)[1]
img=cv.Canny(img,50,150)
```



```
horizontal_kernel = cv.getStructuringElement(cv.MORPH_RECT, (25,1))
vertical_kernel = cv.getStructuringElement(cv.MORPH_RECT, (1, 25));
detected_lines = cv.morphologyEx(img, cv.MORPH_OPEN, horizontal_kernel, iterations=2)
cnts = cv.findContours(detected_lines, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
    cv.drawContours(img, [c], -1, (0,0,0), 2)
detected_lines = cv.morphologyEx(img, cv.MORPH_OPEN, vertical_kernel, iterations=2)
cnts = cv.findContours(detected_lines, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
    cv.drawContours(img, [c], -1, (0,0,0), 2)
```

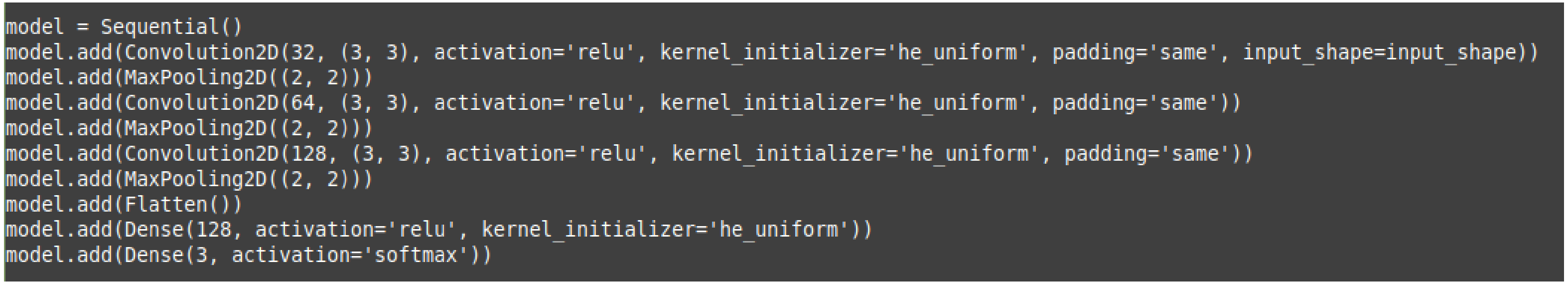
Python Code Part:

1. Create Horizontal and Vertical line kernels.
2. Find horizontal and vertical lines using kernels.
3. And draw horizontal and verical lines in black color over them.

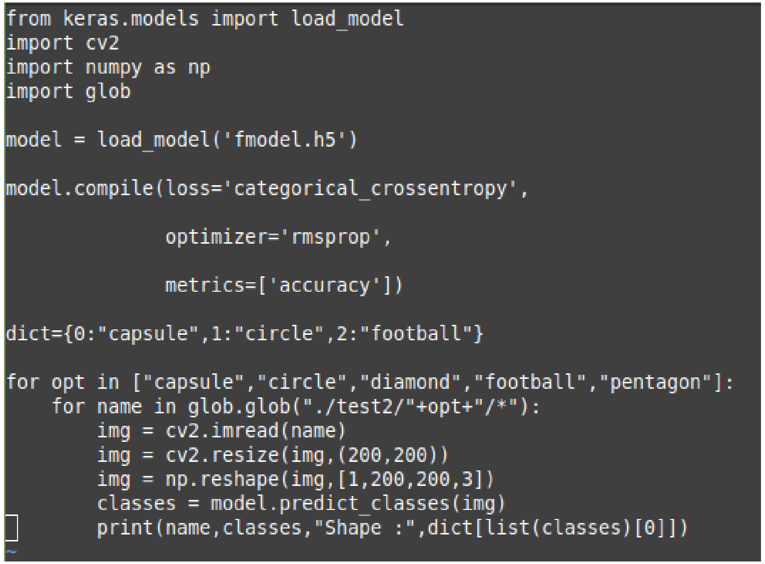
Important Code Parts

Python Code Part:

CNN for model creation



```
model = Sequential()
model.add(Convolution2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same', input_shape=input_shape))
model.add(MaxPooling2D((2, 2)))
model.add(Convolution2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'))
model.add(MaxPooling2D((2, 2)))
model.add(Convolution2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(3, activation='softmax'))
```



```
from keras.models import load_model
import cv2
import numpy as np
import glob

model = load_model('fmodel.h5')

model.compile(loss='categorical_crossentropy',

              optimizer='rmsprop',

              metrics=['accuracy'])

dict={0:"capsule",1:"circle",2:"football"}

for opt in ["capsule","circle","diamond","football","pentagon"]:
    for name in glob.glob("./test2/"+opt+"/*"):
        img = cv2.imread(name)
        img = cv2.resize(img,(200,200))
        img = np.reshape(img,[1,200,200,3])
        classes = model.predict_classes(img)
        print(name,classes,"Shape :",dict[list(classes)[0]])
~
```

Python Code Part:

Image shape prediction using developed model.

THANK YOU